University of Chinese Academy of Sciences

CS091M4042H Assignment 3 — Chapter 4

Pattern Recognition and Machine Learning

熊兴旺*
Sno: 2018E8013261007
中国科学院计算技术研究所

December 31, 2018

*email: xingw.xiong@gmail.com

1 Problem 1

1.1 Problem Description

设有如下三类模式样本集 ω_1，ω_2 和 ω_3，其先验概率相等，求 S_w 和 S_b

$$
\begin{aligned}
\omega_1: &\quad \{(1\ 0)^T, \quad (2\ 0)^T, \quad (1\ 1)^T\} \\
\omega_2: &\quad \{(-1\ 0)^T, \quad (0\ 1)^T, \quad (-1\ 1)^T\} \\
\omega_3: &\quad \{(-1\ -1)^T, \quad (0\ -1)^T, \quad (0\ -2)^T\}
\end{aligned}
$$

1.2 Problem Solution

1. 计算各个类别的均值向量 $\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3$ 和样本总体的均值向量 $\mathbf{m}_0$，

$$
\begin{aligned}
\mathbf{m}_1 &= \frac{1}{3}\left[\begin{pmatrix}1\\0\end{pmatrix} + \begin{pmatrix}2\\0\end{pmatrix} + \begin{pmatrix}1\\1\end{pmatrix}\right] = \begin{pmatrix}\frac{4}{3}\\\frac{1}{3}\end{pmatrix} \\
\mathbf{m}_2 &= \frac{1}{3}\left[\begin{pmatrix}-1\\0\end{pmatrix} + \begin{pmatrix}0\\1\end{pmatrix} + \begin{pmatrix}-1\\1\end{pmatrix}\right] = \begin{pmatrix}-\frac{2}{3}\\\frac{2}{3}\end{pmatrix} \\
\mathbf{m}_3 &= \frac{1}{3}\left[\begin{pmatrix}-1\\-1\end{pmatrix} + \begin{pmatrix}0\\-1\end{pmatrix} + \begin{pmatrix}0\\2\end{pmatrix}\right] = \begin{pmatrix}-\frac{1}{3}\\-\frac{4}{3}\end{pmatrix} \\
\mathbf{m}_0 &= \begin{pmatrix}\frac{1}{9}\\-\frac{1}{9}\end{pmatrix}
\end{aligned}
$$

2. 根据一般特征的散布矩阵准则的公式，可得：

$$
\begin{aligned}
S_w &= \sum_{i=1}^{c} P(\omega_i) E\{(\mathbf{x}-\mathbf{m}_i)(\mathbf{x}-\mathbf{m}_i)^T|\omega_i\} \\
&= \begin{bmatrix}\frac{2}{9} & -\frac{1}{27}\\ -\frac{1}{27} & \frac{2}{9}\end{bmatrix} \\
S_b &= \sum_{i=1}^{c} P(\omega_i)(\mathbf{m}_i-\mathbf{m}_0)(\mathbf{m}_i-\mathbf{m}_0)^T \\
&= \begin{bmatrix}\frac{62}{81} & \frac{13}{81}\\ \frac{13}{81} & \frac{62}{81}\end{bmatrix}
\end{aligned}
$$

3.

1.3 Code Implement

```
#!/usr/bin/env python3

import numpy as np

X1 = np.array([(1,0), (2,0), (1,1)]).T
```

```
X2 = np.array([(-1,0), (0,1), (-1,1)]).T
X3 = np.array([(-1,-1),(0,-1),(0,-2)]).T

X = np.hstack((X1, X2, X3))

m1 = np.mean(X1, axis=1).reshape(-1, 1)
m2 = np.mean(X2, axis=1).reshape(-1, 1)
m3 = np.mean(X3, axis=1).reshape(-1, 1)
m0 = np.mean(X,  axis=1).reshape(-1, 1)

Sw = 1/3 * (np.cov(X1, bias=True) + np.cov(X2, bias=True) + np.cov(X3, bias=True))
print(Sw)

Sb = 1/3 * (np.dot((m1-m0), (m1-m0).T)
        + np.dot((m2-m0), (m2-m0).T)
        + np.dot((m3-m0), (m3-m0).T))

print(Sb)
```

2 Problem 2

2.1 Problem Description

设有如下两类样本集，其出现的概率相等：

$$\omega_1: \quad \{(0\ 0\ 0)^T, \quad (2\ 0\ 0)^T, \quad (2\ 0\ 1)^T, \quad (1\ 2\ 0)^T\}$$
$$\omega_2: \quad \{(0\ 0\ 1)^T, \quad (0\ 1\ 0)^T, \quad (0\ -2\ 1)^T, \quad (1\ 1\ -2)^T\}$$

用 K-L 变换，分别把特征空间维数降到二维和一维，并画出样本在该空间中的位置（可用 matlab 计算）。

2.2 Problem Solution

1. 由于出现的概率相等，于是合并两类样本集，令:

$$X = \begin{bmatrix} 0 & 2 & 2 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & -2 \end{bmatrix}$$

2. 计算矩阵 X 的协方差矩阵 $\Sigma = E[(X-E[X])(X-E[X])^T]$，得：

$$\Sigma = \begin{bmatrix} 0.78571429 & 0.21428571 & -0.10714286 \\ 0.21428571 & 1.35714286 & -0.60714286 \\ -0.10714286 & -0.60714286 & 0.98214286 \end{bmatrix}$$

3. 进行特征分解，求得特征值 $\lambda = (\lambda_1, \lambda_2, \lambda_3)^T$ 和特征向量 $\Phi = (\Phi_1, \Phi_2, \Phi_3)^T$，得:

$$\lambda = \begin{pmatrix} 1.85714286 \\ 0.74144281 \\ 0.52641433 \end{pmatrix}, \Phi = \begin{bmatrix} 0.21538745 & 0.95853318 & -0.18660756 \\ 0.78975397 & -0.05858624 & 0.61061961 \\ -0.57436653 & 0.27889386 & 0.76962413 \end{bmatrix}$$

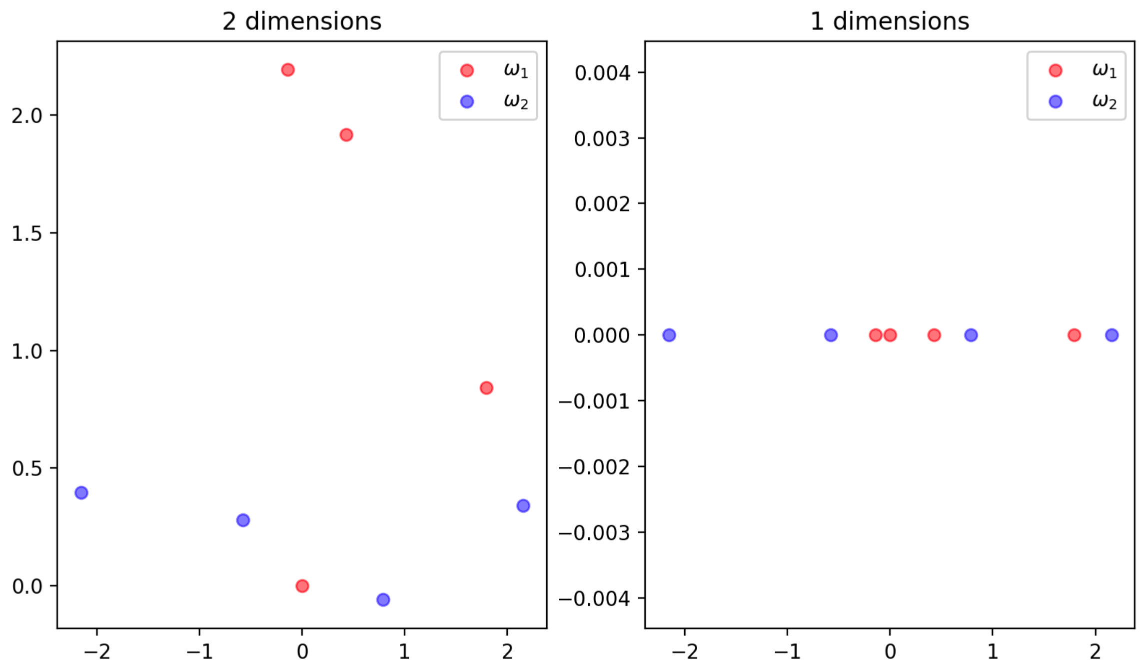
4. 按照特征值的绝对值从大到小进行排序，取最大的前 $k = 2$ 个特征向量，这里取 $\Phi' = (\Phi_1, \Phi_2)^T$，对原来的样本集合进行正交变换，即 $Y = \Phi'^T X$，得到变换后的二维模式特征为：

$$\omega_1: \quad \{(0\ 0)^T, \quad (0.4308\ 1.9171)^T, \quad (-0.1436\ 2.19596)^T, \quad (1.7949\ 0.8414)^T\}$$
$$\omega_2: \quad \{(-0.5744\ 0.2789)^T, \quad (0.7898\ -0.0586)^T, \quad (-2.1539\ 0.3961)^T, \quad (2.1539\ 0.3422)^T\}$$

5. 同理，可以得到变换后的一维模式特征为：

$$\omega_1: \quad \{0, \quad 0.4308, \quad -0.1436, \quad 1.7949\}$$
$$\omega_2: \quad \{-0.5744, \quad 0.7898, \quad -2.1539, \quad 2.1539\}$$

6. 并画出样本在该空间中的位置如下图所示：



2.3 Code Implement

```
#!/usr/bin/env python3

import numpy as np
import matplotlib.pyplot as plt

def pca(X, k):
    cov = np.cov(X)
    eigen_vals, eigen_vecs = np.linalg.eig(cov)
    index = np.argsort(-eigen_vals) # 从大到小
    return eigen_vals[index[:k]], eigen_vecs[:, index[:k]]

X1 = np.array([(0,0,0), (2,0,0), (2,0,1), (1,2,0)]).T
X2 = np.array([(0,0,1), (0,1,0), (0,-2,1),(1,1,-2)]).T
X  = np.hstack((X1, X2))

main_vals, main_vecs = pca(X, 2)
Y1 = np.dot(main_vecs.T, X1)
Y2 = np.dot(main_vecs.T, X2)

print("2 dimensions")
print("Y1:\n", np.around(Y1, decimals=4), "\n")
print("Y2:\n", np.around(Y2, decimals=4), "\n")
```

```python
plt.subplot(121)
plt.scatter(Y1[0], Y1[1], c='red', alpha=0.5, label='$\omega_1$')
plt.scatter(Y2[0], Y2[1], c='blue', alpha=0.5, label='$\omega_2$')
plt.legend()
plt.title('2 dimensions')

main_vals, main_vecs = pca(X, 1)
Y1 = np.dot(main_vecs.T, X1)
Y2 = np.dot(main_vecs.T, X2)

print("1 dimension")
print("Y1:\n", np.around(Y1, decimals=4), "\n")
print("Y2:\n", np.around(Y2, decimals=4), "\n")

plt.subplot(122)
plt.scatter(Y1, np.zeros(Y1.shape), c='red', alpha=0.5, label='$\omega_1$')
plt.scatter(Y2, np.zeros(Y2.shape), c='blue', alpha=0.5, label='$\omega_2$')
plt.legend()
plt.title('1 dimensions')

plt.show()
```